Claims

1. A method for constructing a control flow graph (CFG) from a computer executable program the instructions of which belong to one or more instruction sets, said method comprising the steps of

defining a number of block leader types including at least one type related to an instruction set change, block leaders specifying basic block boundaries in the program, said basic blocks including instructions or data (702),

building a CFG structure comprising basic blocks found in the program (708),

adding control flow and addressing information to said CFG by propagating through said basic blocks and internals thereof (710).

2. The method of claim 1, further comprising a step wherein said CFG is compacted by utilizing a CFG optimisation technique (712).

3. The method of claim 2, wherein said optimisation technique is substantially at least one of the following: unreachable eliminator, branch unconditionalizer, simple redundant eliminator, redundant load eliminator, dead code eliminator, register liveness analyser, branch rationalizer.

4. The method of claim 1, wherein the step of adding control flow and addressing information includes iterating through instructions of a single basic block at a time on the basis of constant propagation information associated with said block.

5. The method of claim 1, wherein the step of adding control flow and addressing information (710) includes propagation of basic block emulation results from a block to another.

6. The method of claim 1, wherein said block leader types further include an entry for at least one of the following: data symbols intermixed with instructions, program entry point, exception vector, relocation entry point, relocation target point, successor of a branch instruction, target of a branch instruction, function.

7. The method of claim 1, wherein the CFG comprises hierarchical levels of sections, functions and basic blocks.

8. The method of claim 1, further comprising the step of reading data from a binary executable file (704).

9. The method of claim 1, further comprising the step of re-constructing an executable from said CFG (714).

10. The method of claim 1, wherein said program is substantially ARM or THUMB architecture specific.

11. A computer program comprising code for carrying out the steps of claim 1 for at least temporary storage in a computer readable medium.

12. A carrier medium for storing a computer executable program for carrying out the steps of claim 1.

13. A system for constructing a control flow graph (CFG) from a computer executable program, said system comprising processing means (806) and memory means (810) for processing and storing instructions and data, and data transfer means (808) for accessing data, said system arranged to define a number of block leader types including at least one type related to an instruction set change, block leaders specifying basic block boundaries in the program, said basic blocks including instructions or data, said system further arranged to build a CFG structure comprising basic blocks found in the program, and to add control flow and addressing information to said CFG by propagating through said basic blocks and internals thereof.

14. The system of claim 13, further arranged to compact the CFG by utilizing a CFG optimisation technique.